8 - Distributed File Systems

Sample

[5 Marks] What is mobility transparency? Provide a high-level conceptual discussion on how you would extend Network File System (NFS) to enable mobility transparency.

Mobility transparency means that if files are moved to another place, no change need to be made on client programs and client administration services.

File group should be added since it allows the file service to be implemented over different servers. Files have UFIDs to guarantee uniqueness across different servers, which helps achieve mobility transparency when files are moved.

Mount service can also be implemented. For example, if a file system is moved to another place, it should be first unmounted from the original place in the remote client, and then be mounted using the new location. The client can always access the files, being unaware of the the changes.

21S1

Q10: Discuss the model architecture of a distributed file system. Illustrate how comprehensive it is by comparing it to the NFS implementation.

DFS architecture: P128

- Client module
- Directory service
- Flat file service

Compare to DFS, NFS has a virtual file system which allows users to access different types of file systems in a uniform way (local and remote). DFS itself is already a comprehensive model.

17S1

Q7: (a) [1 marks] What is typically done to ensure that a UFID is unique across different file servers? P129

To construct a globally unique UFID, we use some unique attribute of the machine on which it is created. e.g. IP address

(b) [1 marks] Why does the interface not provide an Open() primitive?

Because files in the flat file service can be accessed immediately.

(c) [1 marks] Which of the primitives of the interface, if any, are not idempotent? L8 P19

Create() is not idempotent since multiple Create() will create multiple new files.

16S2

Q8: (a) [5 marks] Describe the caching policy used by NFS at the client.

What parameters are used? L8 P31

What checks are done, in what order and why? L8 P35

Tutorial W11

Q1: Name and explain transparencies that should be addressed by distributed file systems.

- Access transparency
    - Client programs don't know if the file if local or remote
- Location transparency
    - Client programs don't know where the file is stored
    - Files can be relocated without changing their pathname
- Mobility transparency
    - Neither client programs nor system administration tables in client nodes need to changed when files are moved
- Performance transparency
    - Maintain acceptable performance while the load on the service varies within a specified range
- Scaling transparency
    - Service can be expanded without loss of performance

Q2: What are the advantages and disadvantages of using absolute names as a naming strategy?

Absolute name provides a complete address to a file including both the server and path names

- machine name: path name

Advantages

- Trivial to find a file once the name is given
- No additional state must be kept since each name is self contained (No global state)
- Greater scalability

- Easy to add and delete new names

Disadvantages

- No location transparency
- File is location dependent and cannot be moved
- Less resilient to failure

Q3: What are the advantages and disadvantages of a naming strategy based on mount points?

Mount Points (used by Sun's Network File System - NFS)

- The client machine creates a set of "local names" which are used to refer to remote locations: mount points
- At boot time, the local name is bound to the remote name
- The operating system must maintain a table to maintain the mapping of what server and path are mapped to each mount point

Advantages

- Names do not contain information about the file location
- Remote location can change between reboots

Disadvantages

- Hard to maintain
  - What happens when machines fail?
  - What happens when files are migrated?
- Can lead to confusion since two different local names may map to the same file on a remote system

Q4: What are the advantages and disadvantages of the following two types of distributed file systems?

A. RPC to access file system calls, no client/local caching

- Advantage: Consistent view of file system
- Disadvantage: Performance (network access, server becomes potential bottleneck)

B. Cache files on clients and perform local file system operations

- Advantage: Local operations = better performance
- Disadvantages
  - What happens when the clients fails?
  - Where should the client store the cached files?
  - Difficult to keep local copy consistent with remote copy